Title: Using models of cortical development based on sparse coding to discriminate between real and synthetically generated faces

Author(s): Nguyen-Fotiadis, Nga Thi Thuy
Moore, Juston Shane
Kenyon, Garrett

Intended for: AIPR, 2020-12-06 (Los Alamos, New Mexico, United States)

Issued: 2022-01-31 (rev.1)

2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR) | 978-1-7281-8243-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/AIPR50011.2020.9425143

Using models of cortical development based on sparse coding to discriminate between real and synthetically-generated faces

Nga T. T. Nguyen
Los Alamos National Laboratory
Los Alamos, NM 87545, USA
nga.nguyen@lanl.gov

Juston S. Moore
Los Alamos National Laboratory
Los Alamos, NM 87545, USA
jmoore01@lanl.gov

Garrett T. Kenyon
Los Alamos National Laboratory
Los Alamos, NM 87545, USA
gkenyon@lanl.gov

***Abstract*—We compare the robustness of image classifiers based on state-of-the-art Deep Neural Networks (DNNs) with classifiers based on a model of cortical development using a single layer of sparse coding. The comparison is based on the ability of the two distinct types of classifiers to distinguish between faces of celebrities from the CelebA dataset and synthetic faces created by the ProGAN multi-scale GAN, trained on the same CelebA images. We examine the robustness of DNNs compared to classifiers based on sparse coding after the addition of universal adversarial perturbations (UAPs), which fool most or all of the DNN classifiers we examined. Our results show that simple classifiers based on sparse coding are robust to UAPs that substantially degrade performance across a wide range of DNN architectures. We hypothesize that sparse latent representations, which correspond to fixed points of a dynamical attractor—or Hopfield network—are naturally denoising and remove small adversarial perturbations. We observe that analogous but reduced robustness is conferred by deep denoising autoencoders. Our results suggest that DNN-based classifiers may be designed to rely on more robust features, and thus may be less susceptible to adversarial attacks, if preceded by a denoising pre-processing layer.**

***Index Terms*—sparse coding, deep convolutional neural network, adversarial examples, robustness, denoising, deepfakes**

I. Motivation

Deep learning has yielded impressive advances across a variety of machine learning tasks such as playing Go at a championship level [1] and correctly labeling natural images [2]. However, highly accurate Deep Neural Networks (DNNs) are susceptible to adversarial attacks that reliably cause misclassification [3]–[5]. Initial demonstrations of adversarial examples used small perturbations that were computed individually for each image to fool a particular DNN classifier. These initial demonstrations were intriguing to the research community because the changes that caused correctly-classified natural images to be misclassified were small enough to be invisible to humans, yet cause state-of-the-art DNNs to misclassify images with high confidence, for example, causing a school bus to be misclassified as an ostrich [3]. Subsequent work demonstrated that universal adversarial perturbations (UAPs) exist that can be added to *any* image in the test set to fool *any* DNN classifier constructed from the same training set [6]. Particularly surprising, UAPs are not class-dependent; rather, one single adversarial pattern can be learned that can when added to *any* image, regardless of class, is likely to cause misclassification. The changes to an image produced by the addition of UAPs are typically larger than those produced by the addition of image-specific adversarial noise. Indeed, published UAPs, although still relatively small, are often readily visible to humans. Published UAPs also typically exhibit clear high-frequency structure, although the resulting patterns are typically semantically meaningless to humans. UAPs likely align with principle directions, corresponding to directions of maximum curvature along decision boundaries [7]. The broad generalizability of UAPs suggests that neural networks are learning "non-robust" features, which are highly dependent on the statistical structure of a specific training dataset [8]. This shortcoming highlights the need for novel approaches to classifier design that improves model robustness.

Unsupervised learning, particularly Boltzmann Machines [9] and sparse coding paradigms [10], seek to learn joint distributions or causes directly from unlabeled data and thus may be less prone to some of the issues that have plagued task-specific deep learning approaches. Moreover, sparse coding has been shown to support near state-of-the-art performance on image labeling tasks using only a linear support vector machine (SVM) classifier [11], [12].

In this paper, we show that a variety of DNN architectures, such as Xception [13], Inception [14], DENSENET [15], and RESNET152 [16], can be trained to differentiate between real and synthetically-generated faces. We then show that each of these DNN classifiers can be fooled by UAPs trained as white box attacks against each architecture separately. To obtain a single set of UAPs that can fool any DNN architecture, we construct multi-layer UAPs [6], [17], which are obtained by successively targeting distinct, state-of-the-art DNN architectures, all of which were initially trained to solve the celebrity/synthetic face discrimination task with high accuracy. After each successive adversarial training stage we obtain a new UAP layer, which corresponds to a new perturbation that is added to a set of images already perturbed

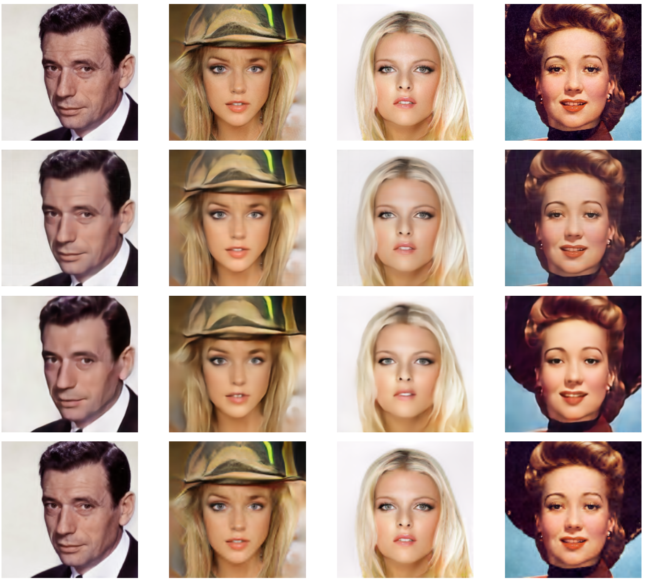

Fig. 1. Top: Examples of cropped, from left to right, real, synthetic, synthetic, and real faces from the study face dataset described in Sec. III. Second: Sparse reconstructions using 512-feature dictionary shown in Fig. 2. Third: Reconstructions by a deep denoising autoencoder. Bottom: Reconstructions by an image smoothing processing. Second row is discussed in detail in Sec. V and Sec. VII, and third and bottom rows in Sec. VII.

by UAPs up to but not including the current layer. Specifically, after each training cycle, we obtain a new UAP that when added to the images in the test set, both better fools the targeted DNN while continuing to transfer to other non-targeted DNNs (white-box and black-box attacks). We then show that classifiers based on one or two cycles of sparse coding are robust to multi-layer UAPs that fool a variety of state-of-the-art DNNs. Finally, we test whether the robustness of classifiers based on sparse coding to multi-layer UAPs arises from their denoising properties. Specifically, we explicitly construct classifiers that employ a deep denoising autoencoder as a front end preprocessor, and show that such classifiers are more robust to the multi-layer UAPs than deep learning classifiers without denoising front-ends but not do not entirely achieve the robustness conferred by 2 cycles of sparse coding.

II. Related Work

One defense strategy against adversarial attacks involves training a deep neural network classifier so as to minimize its classification gradients with respect to small changes in the input image, thereby making the construction of adversarial examples more difficult [18], although artificially amplifying gradients appears to be an effective counter measure [19]. A second defensive strategy entails adding adversarial examples explicitly to the training set [20], [21], but such augmentation-based defenses cannot defend against unseen or novel attacks and often reduces overall classification accuracy. A third defensive strategy seeks to train classifiers to directly detect adversarial examples [22], but as demonstrated by the results presented below, deep neural networks trained to discriminate between real and synthetic faces are themselves susceptible to universal adversarial perturbations whereas classifiers based on one and two cycles of sparse coding are insensitive to identical attacks while retaining high overall accuracy on the same real/synthetic two-class discrimination task. A fourth defense strategy relies on pre-filtering or purifying input images so as to remove adversarial perturbations before passing them to a classifier [23]–[25]. The results presented here not only confirm previous studies showing that adversarial perturbations are often absent from the associated sparse reconstructions [26]–[30] but moreover extend these findings to a task in which the two distributions to be distinguished, corresponding to real and synthetic faces, are optimized to be nearly indistinguishable. In the work presented below, sparse coding is used both to purify the input images and to yield latent representations that support accurate, robust discrimination between real and synthetic faces.

III. Dataset

We study a balanced dataset containing 60,000 face-cropped images of which are 30,000 real celebrity faces and 30,000 synthetic (fake) faces created using multi-scale GANs [31] with input images collected from CelebA [32]. Examples of these real and synthesized faces are shown in the top row of Fig. 1.

We divide the training and test set in each real and synthetic class of images as a random ratio of 5:1, i.e. there are 50,000 face (25,000 of each real and synthetic) images in the training set and 10,000 face (5,000 of each real and synthetic) images in the test set.

IV. Sparse-Coding

The hypothesis that biological neurons encode stimuli by inferring sparse representations explains many of the response properties of simple cells in the mammalian primary visual cortex [10], [33]. Given an overcomplete, non-orthonormal basis $\{\phi_i\}$, inferring a sparse representation involves finding the minimal set of non-zero (positive definite) activation coefficients $\boldsymbol{a}$ that accurately reconstruct a given input signal $\boldsymbol{X}$, corresponding to a minimum of the following energy function:

$$E(\boldsymbol{X},\boldsymbol{\phi},\boldsymbol{a}) = \min_{\{\boldsymbol{a},\boldsymbol{\phi}\}} \left[\frac{1}{2}||\boldsymbol{X}-\boldsymbol{\phi}\boldsymbol{a}||^2 + \lambda||\boldsymbol{a}||_p\right] \quad (1)$$

where λ is a trade-off parameter that determines the balance between the reconstruction error of the original input image $\boldsymbol{X}$ and the number of non-zero (sparse) activation coefficients $\boldsymbol{a} > 0$. A larger λ encourages sparser solutions. This second term in Eq. (1) defines the L_p-norm of the sparsity penalty where $p > 0$. In this paper, we solve Eq. (1) using $p = 1$, corresponding to an L_1-norm. We define $\gamma = \frac{1}{2}\frac{\text{rank}(\boldsymbol{a})}{\text{rank}(\boldsymbol{X})}$ as the overcompleteness of the basis $\{\phi_i\}$, where the factor of $\frac{1}{2}$ arises from the fact that our sparse coefficients are positive definite. γ typically is chosen such that $\gamma \gg 1$ which means that in general there will exist many solutions which achieve a similarly small reconstruction error $||\boldsymbol{X}-\boldsymbol{\phi}\boldsymbol{a}||^2$ and our task

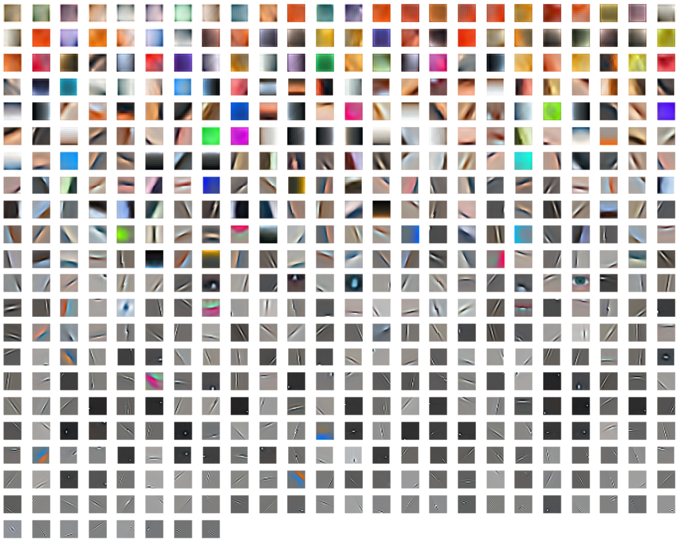

Fig. 2. A complete set of 512 elements of the dictionary, learned from 25,000 images of real faces, sorted by their appearance frequency.

is to find the sparsest one. Both the feature maps $\boldsymbol{a}$ and the dictionary of feature kernels $\boldsymbol{\phi}$ can be determined by a variety of standard methods. Here, we solved for the feature maps using a convolutional generalization, previously described [12], [34], of the Locally Competitive Algorithm (LCA) [35], where the feature kernels themselves are adapted according to a local Hebbian learning rule that reduces reconstruction error given a sparse representation. Dictionary learning was performed via Stochastic Gradient Descent (SGD) using only real celebrity faces drawn from the training data set. Both dictionary learning and sparse coding was performed using PetaVision [36].

V. Optimizing a Dictionary for Sparse Coding of the CelebA Dataset

We trained a convolutional dictionary for sparse reconstruction of the 25,000 224 × 224 CelebA images of celebrity faces in the training set. Each feature kernel was replicated with a stride of 2 pixels in both the vertical and horizontal directions, resulting in a feature map of size $112 \times 112 \times 512$, corresponding to an overcompleteness of $\gamma = 21.3$. The set of 512 learned feature kernels can be visualized as RGB color image patches 18 × 18 in Fig. 2, shown in rank order of activation frequency. The average activation frequency was approximately 1%. To reconstruct the original image (as input $\boldsymbol{X}$), we generate the sparse representation $\boldsymbol{a}$ obtained by optimizing Eq. (1) using learned dictionary $\boldsymbol{\phi}$ on the 25,000 real face images. Some examples of RGB images of sparse reconstruction are shown as the second row in Fig. 1.

VI. Universal Adversarial Perturbation

A. Universal adversarial attack

[6] and [17] showed that state-of-the-art DNNs can be fooled with a single perturbation, independent of the input images, built upon universal adversarial networks (UANs). Based on a generative model architecture, an Universal Adversarial Perturbation (UAP) is crafted from an UAN that generates perturbations from sampled noise. When this perturbation is added to the input image, the targeted neural network is more likely to misclassify the input object. UANs typically consists of several deconvolutional layers with activation ReLUs followed by several fully connected tables. The UAP is optimized [5], [17], [37] by the following objective function:

$$L_{\text{UAP}} = \max\left\{\log\left[f(\boldsymbol{X}+\boldsymbol{\delta})\right]_{c_0} - \max_{i \neq c_0} \log\left[f(\boldsymbol{X}+ \boldsymbol{\delta})\right]_i - \kappa\right\} + \alpha \left\|\boldsymbol{\delta}\right\|_p \tag{2}$$

where $\boldsymbol{X}$ and $\boldsymbol{\delta}$ are the image input and scaled noise, respectively. α is a control parameter on the noise distance loss [last term in Eq. (3)]. f is a probability (e.g. softmax) function. The UAP was minimized over the training celebrity face images until a confidence criteria κ is reached. We employ the $L_{\infty(p=\infty)}$-norm to optimize the loss in Eq. (3). We stop the optimization (see also Ref. [17]) when classifier $f(X+\delta)$ misclassifies $\boldsymbol{X}$ of class i as class c_0, where $c_0 \neq i$, such that the largest single perturbation in $\boldsymbol{\delta}$ is minimal. Here, the classes are binary (real and fake), thus i and c_0 are $\in \{\text{real}, \text{synthetic}\}$.

B. Deep neural networks (DNNs)

We study five DNN models: RESNET50, RESNET152 [16], DENSENET121 [15], Inception-v3 [14] and Xception [13].

All five networks were optimized on the same 50,000 “clean” (no-adversarial-noise) training set and all generalized perfectly to the 10,000 test faces in the holdout dataset which each DNN architecture achieving a perfect classification score (absolute zero error) as shown in Table I. Perfect classification scores on this task are not entirely surprising if we recall that the synthetic images are generated using a multi-scale GAN. By allowing the different DNN architectures to optimize on a fixed distribution, we are in essence fixing the generator half of a GAN so that only the detector half of the GAN can adapt.

Next, all five DNN classifiers are tested on the 10,000 “noisy” test faces. We first attack each of the five deep learning architectures separately. Specifically, we create separate UAPs for each of the 5 DNN architectures which we use to create 5 ’noisy’ test sets and examine performance of each DNN classifiers on the noisy test sets optimized for each architecture. The transferability of these five separately optimized attacks on the other four remaining networks is summarized in Table II. Transferability, measured by the off-diagonal elements in Table II, attains the largest values between the pair of DNNs **RESNET152** (being targeted, column label in Table II) and nontargeted RESNET50 (row label in Table II), which is $T_{(\textbf{RESNET152},\text{RESNET50})} = 50.04\%$, and also between $T_{(\textbf{Xception},\text{Inception-v3})} = 50.16\%$. Low transferability is found with $T_{(\textbf{DENSENET121},\text{RESNET152})} = 22.93\%$ or $T_{(\textbf{RESNET152},\text{DENSENET121})} = 24.5\%$. Note that classification accuracy is given by $100\% - T$, i.e. the higher the success attack rate (classification error), the lower the accuracy of the targeted classifier. In this paper, we generated UAPs that were approximately 4% of the input image, i.e. ($\frac{\|\boldsymbol{\delta}\|_p}{\|\boldsymbol{X}\|_p} = 0.04$). Xception and RESNET50 were the most fragile DNNs (attacks optimized for these architectures were least transferable to other DNNs) with a maximum $T_{(\mathbf{A},\mathrm{B})} \approx 50\%$ where $\mathbf{A}$ = **Xception** or **RESNET50** and B being the remaining 4 DNNs in each case) while DENSENET is the least fragile DNN of 5 architectures explored here. In Table II, transferability from attacking any one in the 5 DNNs to the remaining 4 architectures reaches a highest value of about 50%.

C. Multiple-layer universal adversarial perturbations (UAPs)

To obtain a single adversarial perturbation that would be effective against any DNN architecture, we created a multi-layer UAP by attacking different DNN architectures successively. Here, a second layer of UAPs is generated by starting with a set of test images to which UAPs targeted against Xception have been added and then performing a second adversarial attack against RESNET50. A third layer of UAPs is then generated by a subsequent attack against DENSENET121. Adversarial perturbations obtained by these sequential attacks are shown in Fig. 3. We explored a large number of possible multi-layer UAP sequences and determined the sequence employed here to be the most effective.

For the {celebrity,synthetic} face discrimination task, we find that multi-layer UAPs can be constructed which transfer well between different DNN architectures. As shown in Table III, we obtain transferability above 45% between any pair of three targeted networks (Xception, RESNET50, DENSENET121). Although constructed by successive attacks, multi-layer UAPs nonetheless represent a single, universal perturbation procedure that can be applied to any test image in order to fool a wide-range of DNN architectures. An example of a test image to which the 3 layers of UAPs (right panel in Fig. 3) have been added is shown as the second panel from the left of Fig. 4. Classification error that translates as attack success rate (or transferability, applicable to networks that are not being attacked) for all 5 networks is summarized in Table III. Corresponding classification accuracy ($100\% - T$) from Table III reads 22.98%, 54.82%, 23.66%, 49.97%, and 16.52% for RESNET50, RESNET152, DENSENET121, Inception-v3, and Xception, respectively. Note that only RESNET152 has a classification accuracy above chance (50%). We regularized the 2nd and 3rd layer UAPs (optimized against RESNET50 and DENSENET121, respectively) by requiring that the attack success rate for each of the 5 DNNs must not decrease considerably (accepted fluctuations $< 3\%$ if decreased) when another new UAP layer is added. This regularization explains the low classification accuracy for the three targeted networks, particularly Xception (16.52%), for the the 3-layer UAP, because Xception is the first layer of this multiple layer attack with an attack success rate of 80.17% (last diagonal element in transferability matrix in Table II, equivalent 19.83% accuracy) and this number was increased after two additional layers of UAPs optimized against RESNET50 and DENSENET121, respectively.

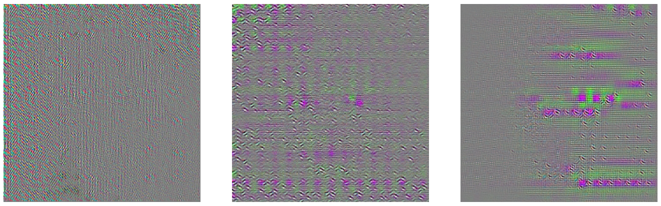

Fig. 3. One, two and three layer UAPs generated by attacking, from left to right, Xception, Xception and RESNET50, and Xception and RESNET50 and DENSENET121, respectively.

VII. Robustness of Deep Neural Networks vs. Sparse Coding against Universal Adversarial Perturbations (UAPs)

A. Denoisers

We compare three denoising strategies. (i) One of the most widely used applications of sparse coding is denoising [38]. (ii) To make a fair comparison between DNN and sparse coding based classifiers, we added a “pre-processing” denoising stage to the DNN, which was then retrained and applied to the UAP test set derived from the original DNNs. The denoising autoencoder consisted of 3 convolutional layers with 128, 64 and 32 kernels. (iii) We used another denoiser that is an image smoothing process based on a wavelet decomposition protocol

	RESNET50	RESNET152	DENSENET121	Inception-v3	Xception
Classification error (%) (no noise)	0.0	0.0	0.0	0.0	0.0

TABLE I

CLASSIFICATION ERROR OF ALL FIVE STUDIED DEEP NEURAL NETWORKS.

	RESNET50	RESNET152	DENSENET121	Inception-v3	Xception
RESNET50	**80.03**	47.92	40.44	33.09	49.93
RESNET152	50.04	**87.01**	24.5	49.98	49.93
DENSENET121	50.02	22.93	**70.09**	37.31	49.36
Inception-v3	49.94	46.09	47.68	**80.11**	49.02
Xception	49.9	43.28	49.98	50.16	**80.17**

TABLE II

SUMMARY OF WHITE-BOX ATTACK SUCCESS AND TRANSFERABILITY (%) OF UAPS TARGETED AGAINST EACH OF FIVE DNN MODELS OF TABLE I. BOLD TEXTS INDICATE ATTACKED NETWORKS. BOLD NUMBERS (DIAGONAL ELEMENTS) HIGHLIGHT THE WHITE-BOX ATTACK RATES OF SUCCESS ON THE CORRESPONDING DNNS LISTED IN THE ASSOCIATED COLUMN. OFF-DIAGONAL ELEMENT MEASURES HOW WELL THE UAP TRANSFERS TO EACH OF THE OTHER 4 DNN MODELS WITH HIGHER VALUES DENOTING BETTER TRANSFERABILITY.

	RESNET50	RESNET152	DENSENET121	Inception-v3	Xception
Classification error (%) (3 UAP-layer noise)	77.02	45.18	76.34	50.03	83.48

TABLE III

CLASSIFICATION ERROR ON THE NOISY TEST SET WHERE NOISE SET IS OBTAINED BY ATTACKING SUCCESSIVELY XCEPTION AND RESNET50 AND DENSENET121 FOR THE 5 STUDIED DNNS IN TABLE VI-B.

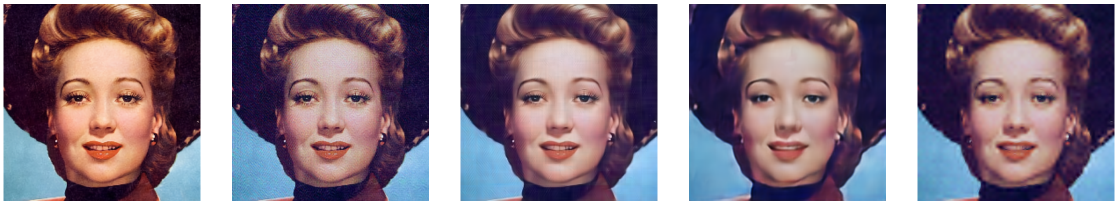

Fig. 4. Example of a test image to which layers of UAPs have been added, corresponding to successive attacks against the combination of 3 noise type generated using Xception followed by RESNET50 and then followed by DENSENET121. From left to right: original test example, study UAP-added test example, test example denoised by sparse coding, test example denoised by deep autoencoder, and test example denoised with an image smoothing processor.

(as an example see [39]). Here, we generated each new 60,000 dataset from the no-adversarial-noise 60,000 faces for each case of (i) sparse coding, (ii) denoising autoencoder, and (iii) smoothing. Each separate 10,000 image test set input for each denoising procedure was also generated that "denoised" the test faces with 3 UAP layers of noises attached. As a result, each denoiser now outputs a new different 10,000 denoised image test set. Mean-square error (MSE) is evaluated as the mean square distance from $N = 10,000$ denoised test images $\boldsymbol{X}^i_{\text{recon}}$ of length M to their input images $\boldsymbol{X}^i$ as $\frac{1}{NM}\sum_i \|\boldsymbol{X}^i - \boldsymbol{X}^i_{\text{recon}}\|^2$). These 3 denoisers (i), (ii), and (iii) generate output images that look as, respectively, second, third, and bottom row in Fig. 1 for the case with no adversarial noises added and corresponding MSE being 109.86 (i), 111.57 (ii), and 37.54 (iii).

B. Robustness from denoising

We trained separate classifiers on each of the datasets generated using the 4 denoising strategies. (i) A linear Support Vector Machine (SVM) [40] for classifying 4×4 max-pooled sparse representations. An Xception model for classifying the output of either the (ii) deep denoising autoencoder (AE), (iii) smoothing filter, or (iv) sparse reconstructions, respectively. Corresponding classification accuracy for the different classifiers applied to the non-noisy test dataset are shown as blue columns in Fig 5. In the absence of 3-layer UAPs, all 5 DNNs attain perfect scores on the holdout test dataset (see bottom row in Tables I. Xception, which was highly susceptible to the multi-layer UAP attack, was used for assessing the impact of the denoising pre-processing stage. All 4 denoising classifiers (both linear SVM and Xception based) do well on the hold out test dataset in the absence of 3-layer UAP holdout test dataset with classification accuracy all above 98%. For ease of comparison, we replot the performance of the Xception classifier (shown previously in Table I). The first cycle of pooled sparse latent representation passed to a linear SVM yielded 98.57% accuracy. To achieve additional denoising, we performed a 2nd cycle of sparse coding on the sparse reconstructions generated from the 1st cycle and passed the resulting

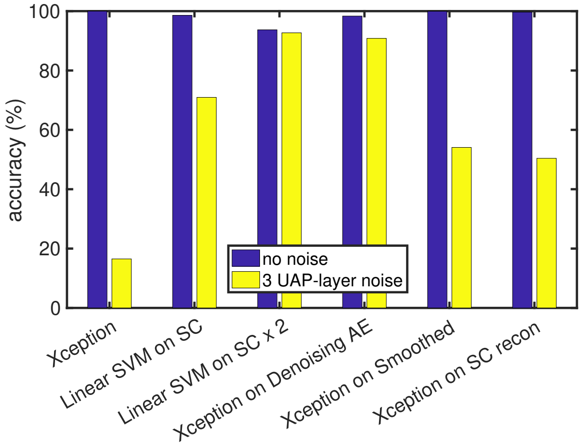

Fig. 5. Classification accuracy with and without a denoising pre-processing stage. Blue and yellow columns refer to holdout test images with and without the addition of 3-layer UAPs, respectively. First column set) Xception without any denoising pre-processing stage. Second and third column sets) A linear SVM applied to the 4×4 max-pooled latent representations resulting from one and two cycles of sparse coding (SC), respectively. Fourth, fifth and sixth column sets) An Xception model was trained on images passed through one of three denoising pre-processing stages, either a denoising autoencoder (AE), a smoothing filter, or a sparse reconstruction (recon). UAPs were not present in any of the training images with or without a denoising pre-processing stage. Two cycles of sparse coding produced the most robust classifier, followed closely by a denoising autoencoder.

4×4 max-pooled sparse latent representations to a retrained linear SVM, yielding a reduced accuracy of 93.71% on the original holdout test dataset in the absence of 3-layer UAPs. Xception preceded by a deep denoising autoencoder yielded an accuracy of 98.34%, Xception preceded by smoothed images produced an accuracy of 99.96%, and finally Xception applied to sparse reconstructions yielded an accuracy of 99.68%. Note again that all classifiers discussed in this paper are trained on the same 50,000 training image datasets in the absence of any added multi-layer UAPs.

The sparse reconstruction of a noisy image from our test dataset to which the 3-layer UAP has been added is shown in Fig. 4 (third panel from the left). Analogous denoised output images for the same noisy image with 3-layer UAP added using either a deep denoising autoencoder or a smoothing filter are shown as the fourth and fifth panels in Fig. 4, respectively. Mean square distance, MSE, of the 10,000 denoised test images to their "*noisy*" inputs, respectively, is 95.2 (i), 153.81 (ii), and 118.72 (iii), confirming visual inspection which reveals that the denoising autoencoder most completely removes the added 3-layer UAP, although the added UAPs are significantly removed by all three denoising pre-processing stages. As noted previously, adding 3-layer UAPs to the image test dataset drastically reduced the accuracy of the Xception classifier trained on the original non-noisy training dataset, as shown by the first yellow column in Fig. 5, producing a classification accuracy on the adversarial test dataset of only 16.52%. Classification results on the adversarial test dataset obtained using one and two cycles of sparse coding, pooled to a 4×4 grid and combined with a linear SVM trained on the non-adversarial training set achieves 70.95% and 92.69%, respectively, as shown by the second and third yellow columns in Fig. 5). The accuracy on the adversarial holdout test dataset obtained by applying the DNN Xception classifiers trained separately on the non-adversarial training dataset passed through each of the the three denoising pre-processors, (i) denoising autoencoder (AE), (ii) smoothing filter, and (iii) sparse reconstructions, shown by the last 3 yellow columns in Fig. 5, was lower, 90.82%, 54.08%, and 50.42%, respectively, although the denoising autoencoder was only slightly lower. The above results show that without a pre-processing denoising stage, both one and two cycles of sparse coding followed by a linear SVM outperforms an Xception model (and the other 4 DNNs, see also Table III) on the adversarial test dataset when trained on the non-adversarial training set. However, after a deep denoising autoencoder is added as a pre-processing step, the Xception model improves substantially in classification performance, almost matching the classification performance achieved by 2 cycles of sparse coding followed by a linear SVM. This substantial improvement in the classification performance of the Xception model when preceded by a deep denoising autoencoder is no longer seen in case we used a different type of denoising pre-processing stage, i.e sparse reconstruction or smoothing. As noted above, both sparse reconstruction and smoothing filters preserve the added 3-layer UAPs more than does the denoising autoencoder, perhaps explaining why the denoising autoencoder yields a more robust classifier when employed as a pre-processing stage. Classifiers based on Xception all perform very well on non-adversarial holdout test images, whether the original (non-adversarial) training images are passed through a deep denoising autoencoder, sparse reconstruction or smoothing filter. However, the resulting classifiers are different with respect to their sensitivity to 3-layer UAPs. A classifier based on a linear SVM applied to the pooled latent representations resulting from 2 cycles of sparse coding are almost completely resistant to the addition of the 3-layer UAPs, suggesting that such classifiers are using different criteria for the real vs synthetic discrimination task.

VIII. Discussion

The ability to detect synthetic content will become increasingly important in order for society to distinguish between real and fake news sources. Our results confirm that DNNs can reliably detect synthetic content in fixed databases. However, DNNs are intrinsically susceptible to being fooled by adversarial examples. Indeed, GANs work by explicitly constructing adversarial examples. Thus, using DNNs to detect GAN-generated synthetic images is inherently circular. Here, we show that classifiers based on a single layer of sparse coding optimized for the reconstruction on celebrity faces can detect synthetic faces with high accuracy in a manner that is robust to conventional adversarial attacks, a robustness that arises from the denoising properties of sparse reconstructions. A linear SVM classifier is able to produce accurate discriminations between celebrity and synthetic faces based on their sparse latent representations, suggesting that the sparse latent representation averaged over all celebrity faces provides a principle direction against which synthetic faces can be compared.

IX. Acknowledgements

Research presented in this paper was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project numbers 20180742ER, 20200666DI, and 20210043DR.

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, “Mastering the game of Go without human knowledge,” Nature, vol. 550, pp. 354–359, October 2017.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and Li Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” arXiv:1409.0575, September 2014.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” arXiv:1312.6199, December 2013.

[4] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” in Proceedings of the 3rd International Conference on Learning Representations ICLR, 2015.

[5] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in Security and Privacy (SP) IEEE Symposium, 2017, pp. 39-–57.

[6] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, 1765–1773.

[7] S. Jetley, N. Lord, and P. Torr, “With friends like these, who needs adversaries?” in Advances in Neural Information Processing Systems, 2018, pp. 10749–10759.

[8] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” in Advances in Neural Information Processing Systems, 2019, pp. 125-136.

[9] G. E. Hinton, “A Practical Guide to Training Restricted Boltzmann Machines,” in Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol. 7700, Springer, Berlin, Heidelberg, 2012, pp. 599–619.

[10] B. A. Olshausen and D. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” vol. 381, pp. 607–609, June 1996.

[11] A. Coates and A. Y. Ng., “The Importance of Encoding versus Training with Sparse Coding and Vector Quantization,” in Proceedings of The 28th International Conference on Machine Learning (ICML), 2001, pp. 921–928.

[12] Z. Zhang and G. T. Kenyon, “A Deconvolutional Strategy for Implementing Large Patch Sizes Supports Improved Image Classification,” in First International Workshop on Computational Models of the Visual Cortex: Hierarchies, Layers, Sparsity, Saliency and Attention, 2015.

[13] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” arXiv:1610.02357, October 2016.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2818–2826.

[15] G. Huang, Z. Liu, L.v. der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” arXiv:1608.06993, August 2018.

[16] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” arXiv:1512.03385, December 2015.

[17] J. Hayes and G. Danezis, “Learning Universal Adversarial Perturbations with Generative Models,” in Proceedings of IEEE Security and Privacy Workshops (SPW), 2018.

[18] N. Papernot, P. McDaniel, X. Wu, S. Jha and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” 2016 IEEE Symposium on Security and Privacy (SP), pp. 582–597, 2016.

[19] N. Carlini and D. Wagner, “Defensive distillation is not robust to adversarial examples,” arXiv preprint arXiv:1607.04311 2016.

[20] I. J Goodfellow, J. Shlens and C. Szegedy, , “Explaining and harnessing adversarial examples,” arXiv preprint arXiv:1412.6572 2014.

[21] T. Miyato, S. Maeda, M. Koyama, K. Nakae and S. Ishii, “Distributional smoothing with virtual adversarial training,” arXiv preprint arXiv:1507.00677 2015.

[22] Z. Zheng and P. Hong, “Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks,” in Advances in Neural Information Processing Systems, 2018, pp. 913–7922.

[23] U. Hwang, J. Park, H. Jang, S. Yoon and N. I. Cho, “Puvae: A variational autoencoder to purify adversarial examples”, IEEE Access, vol. 7, pp. 126582–126593, December 2019.

[24] Y. Song, T. Kim, S. Nowozin, S. Ermon and N. Kushman, “Pixeldefend: Leveraging generative models to understand and defend against adversarial examples”, arXiv:1710.10766, 2017.

[25] D. Meng and H. Chen, “MagNet: A two-pronged defense against adversarial examples”, Proc. ACM SIGSAC Conf. Comput. Commun. Secur., pp. 135-147, 2017.

[26] B. Sun, N. Tsai, F. Liu, R. Yu and H. Su, “Adversarial defense by stratified convolutional sparse coding”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 11447–11456, 2019.

[27] E. Kim, J. Yarnall, P. Shah, and G. T. Kenyon. “A Neuromorphic Sparse Coding Defense to Adversarial Images.” In Proceedings of the International Conference on Neuromorphic Systems, pp. 1-8. 2019.

[28] J. M. Springer, C. S. Strauss, A. M. Thresher, E. Kim, and G. T. Kenyon, “Classifiers Based on Deep Sparse Coding Architectures are Robust to Deep Learning Transferable Examples”, arXiv preprint arXiv:1811.07211, 2018.

[29] E. Kim, J. Rego, Y. Watkins, and G. T. Kenyon, “Modeling Biological Immunity to Adversarial Examples”, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4666-4675, 2020.

[30] D. M. Paiton, “Analysis and applications of the Locally Competitive Algorithm.” PhD diss., UC Berkeley, 2019.

[31] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” arXiv:1710.10196, October 2017.

[32] Large-scale CelebFaces Attributes (CelebA) Dataset, http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html.

[33] B. A. Olshausen and D. Field, “Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1?” Vision Res., vol. 37, pp. 3311–3325, December 1997.

[34] P. F. Schultz, D. M. Paiton, W. Lu, and G. T. Kenyon, “Replicating kernels with a short stride allows sparse reconstructions with fewer independent kernels,” arXiv:1406.4205, June 2014.

[35] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen, “Sparse Coding via Thresholding and Local Competition in Neural Circuits,” Neural Computation, vol. 20, pp. 2526–2563, October 2008.

[36] PetaVision, https://petavision.github.io.

[37] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “ZOO: Zeroth Order Optimization based Black-box Attacks to Deep Neural Networks without Training Substitute Models,” arXiv:1708.03999, August, 2017.

[38] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” IEEE Transactions on Image processing, vol. 15, pp. 3736–3745, December 2006.

[39] Y. Ting-Hua, L. Hong-Nan, and Z. Xiao-Yan, “Noise Smoothing for Structural Vibration Test Signals Using an Improved Wavelet Thresholding Technique,” Sensors, vol. 12, pp. 11205–11220, August 2012.

[40] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” ACM Transactions on Intelligent Systems and Technology, vol. 2, pp. 27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.